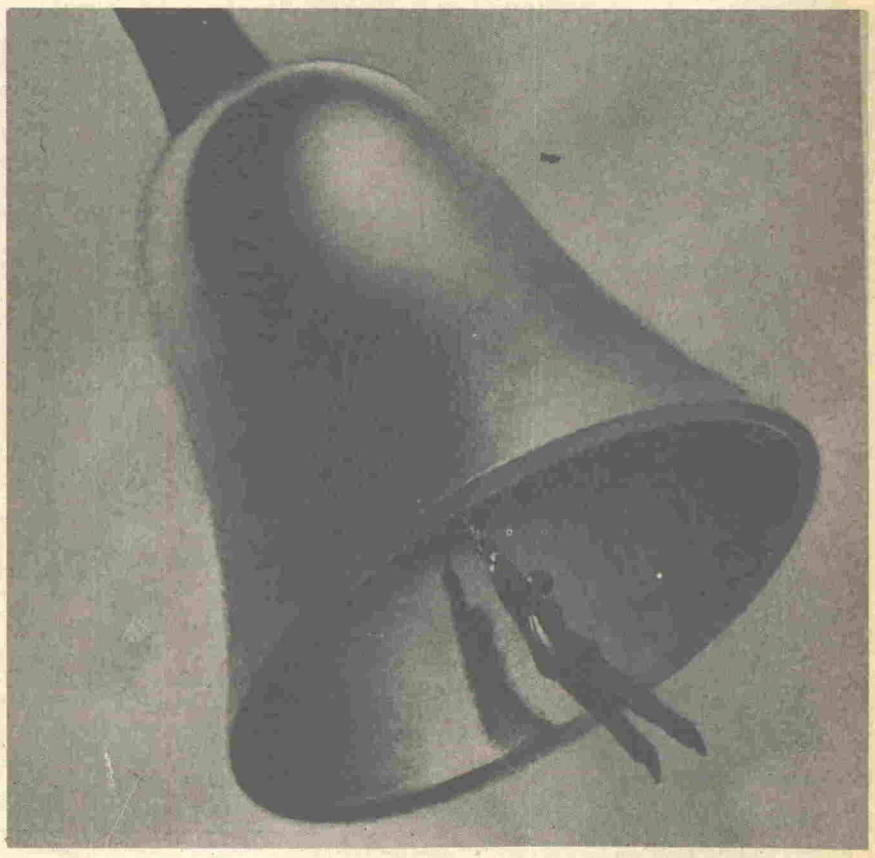
ซอฟต์แวร์
บุญเลิศ เอี่ยมพัศนา

SOFT MAIL

TPW
: ไฟล์ไดอะล็อกบ็อกซ์กลาง

ทำความรู้จักกับ
ไดอะล็อกบ็อกซ์กลาง
เราแจกแจงรายละเอียดต่าง ๆ
ไว้ให้ผู้อ่านได้ศึกษาเพิ่มเติม



วินโดวส์ 3.1 อาจจะมีความหมายหนึ่งสำหรับผู้ใช้ แต่สำหรับผู้เขียนโปรแกรมวินโดวส์ 3.1 ก็คือขบวนรถเร็วครับ ที่ช่วยให้เขียนโปรแกรมได้เร็วขึ้น (จึงสงวนคำว่าขบวนรถด่วนไว้ก่อน) สิ่งหนึ่ง (โบกี้หนึ่ง) ก็คือไดอะล็อกบ็อกซ์กลาง (Common Dialog Box Library) ซึ่งผมเองก็อึดอัดใจที่จะนำมากล่าวในช่วงนี้ แต่เมื่อมีผู้ถามไปก็ขอนำมาเล่าสู่กันฟังครับ

โดยจะนำมาใช้กับโครงงานจับภาพใส่หน้าต่างที่ตั้งใจไว้เดิม โปรแกรมนี้จึงแบ่งออกเป็น 3 ตอน 3 ฉบับ (ซึ่งควรจะติดต่อกัน) คือ

– ไดอะล็อกบ็อกซ์กลาง กล่าวถึงการใช้ไดอะล็อกบ็อกซ์สำเร็จรูปของวินโดวส์

– จับภาพใส่หน้าต่าง กล่าวถึงการลอกภาพจากจอภาพไปให้หน้าต่างและระหว่างบิตแม็พ

– ติดต่อผ่านคลิปบอร์ด กล่าวถึงการแลกเปลี่ยนข้อมูลระหว่างแอพพลิเคชันผ่านทางคลิปบอร์ด

รวมกันแล้วจะเป็นโปรแกรม ScreenCaptor ซึ่งต้องถือว่าเริ่มตั้งแต่ฉบับนี้

ที่ถามไป

มีผู้ถามปัญหาไป เป็นเรื่องที่ผมเห็นว่าควรนำมาตอบให้ผู้อ่านได้ทราบทั่วกัน คือ

– ให้กล่าวถึงการใช้รีซอร์สเวอร์กชอบของบอร์แลนด์บ้าง จะนำมากล่าวถึงครับ

– ทำอย่างไรจึงจะใช้ไดอะล็อกบ็อกซ์สวย ๆ ของวินโดวส์ได้ ก็คือบทความตอนนี้

– ให้กล่าวถึงการใช้ Borland Pascal with Objects 7.0 บ้าง ผมเองก็ยังไม่ได้ใช้จริงจังครับ โปรแกรมชุดนี้มีคอมพายเลอร์อยู่ 2 กลุ่ม กลุ่มหนึ่งใช้สำหรับรันบนดอส อีกกลุ่มหนึ่งใช้สำหรับรันบนวินโดวส์ มีไอคอนชื่อ BPW ซึ่งโปรแกรมที่ผมเขียนโดยใช้ TPW สามารถใช้กับ BPW ได้ด้วย แต่ต้องเปลี่ยนการกำหนดใช้ยูนิตใน USES ใหม่ ดังรูปที่ 6 อยู่ตอนท้ายครับ

ขอบคุณทุกท่านครับที่ถามไป ทำให้ผมได้นำเวอร์กชอบกับไดอะล็อกบ็อกซ์กลางมาใช้ ไม่เช่นนั้นคงเผลอลืมไปเลย

ไดอะล็อกบ็อกซ์กลาง

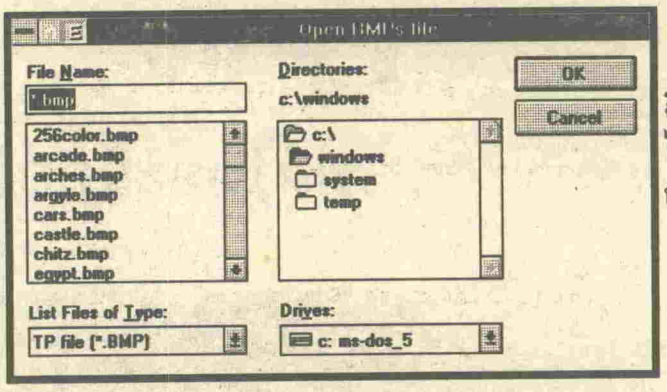
ขบวนรถเร็วที่ผมเปรียบเทียบเป็นโบกี้หนึ่งคือ ไลบรารีหนึ่งมีอยู่หลายไลบรารี Common Dialog Box Library คือไลบรารีหนึ่ง มีไดอะล็อกบ็อกซ์อยู่หลายแบบ โดยที่ในวินโดวส์ได้ทำเป็นฟังก์ชันไว้ และในเทอร์โบปาสคาลได้นำมาทำเป็นยูนิต CommDlg ซึ่งไม่มีในคู่มือต้องดูใน Help มีฟังก์ชันดังนี้

- ChooseColor
- ChooseFont
- FindText และ ReplaceText
- GetFileTitle
- GetOpenFileName และ GetSaveFileName
- PrintDlg

ผมคิดว่าชื่อคงจะบ่งบอกหน้าที่เป็นอย่างดีครับ เช่น เมื่อเรียกใช้ ChooseColor มันจะแสดงไดอะล็อกบ็อกซ์ให้ผู้ใช้ผสมสี ในคู่มือของวินโดวส์กล่าวไว้ว่า มีวิธีใช้ใกล้เคียงกัน จึงจะนำ GetOpenFileName กับ GetSaveFileName และ PrintDlg มาแนะนำครับ

GetOpenFileName

เมื่อเรียกใช้ฟังก์ชันนี้ มันจะแสดงไดอะล็อกบ็อกซ์ดังรูปที่ 1 ซึ่งจะเป็นไปตามค่าที่ผ่านไปให้ด้วย และค่าที่ผ่านไปนี้ เป็นเรื่องสำคัญเพียงเรื่องเดียวสำหรับการใช้ฟังก์ชันในชุดนี้ โดยได้ทำเป็นแบบข้อมูลเรคอร์ดไว้ ของ GetOpenFileName และ GetSaveFileName คือ TOpenFileName ตามรูปที่ 2 แต่โดยทั่วไปไม่ได้ใช้ทั้งหมด เพราะออกแบบไว้สำหรับการใช้งานที่กว้างขวางมาก ซึ่งผมจะขออธิบาย



รูปที่ 1
ไดอะล็อกบ็อกซ์กลางของฟังก์ชัน GetOpenFileName

```
TOpenFilename = record
  lStructSize       : Longint;
  hWndOwner         : HWnd;
  hInstance         : THandle;
  lpstrFilter       : PChar;
  lpstrCustomFilter: PChar;
  nMaxCustFilter    : Longint;
  nFilterIndex      : Longint;
  lpstrInitialDir   : PChar;
  lpstrFile         : PChar;
  nMaxFile          : Longint;
  lpstrFileTitle    : PChar;
  nMaxFileTitle     : Longint;
  lpstrTitle        : PChar;
  Flags             : Longint;
  nFileOffset       : Word;
  nFileExtension    : Word;
  lpstrDefExt       : PChar;
  lCustData         : Longint;
  lpfnHook : function (Wnd: HWnd; Msg,
    wParam: Word; lParam: Longint): Word;
  lpTemplateName    : PChar;
end;
```

รูปที่ 2
แบบข้อมูล TOpenFileName

ลิสติงที่ 1

```
PROGRAM  TestFileOpen;  { fiopen }
USES   commdlg,WinTypes,WinProcs,WObjects,Strings,WinCrt;

CONST  appname  = 'fiopen';
       thetitle = 'TEST File Open';
       menursrc = 'SCRNCAP2_MENU';

TYPE
  WndProcP = ^WndProcO;
  WndProcO = OBJECT(TWindow)
    curfi : array[0..79] of Char;
 constructor  Init(apar : PWindowsObject; atitle : PChar);
 procedure  wmLButtonDown(var msg : TMessage);
             virtual wm_First + wm_LButtonDown;
            END;


(* ------------------ WndProcO Routine -------------- *)
```

ลิสติงที่ 1 (ต่อ)

```
CONSTRUCTOR  WndProC.Init(apar : PWindowsObject; atitle : PChar);
BEGIN
  TWindow.Init(apar,atitle);
  StrCopy(curfi, 'c:\windows\*.bmp');
END;


CONST  numfld = 10;
       filtarray : array[1..numfld] of PChar =
        ( 'Bitmap file (*.BMP)', '*.bmp',
          'Pascal file (*.PAS *.BAK)', '*.pas;*.bak',
          'Document file (*.DOC)', '*.doc',
          'Header file (*.H)', '*.h',
          'All files (*.*)', '*.*');

CONST
  oldfltr : Word = 1;
  userflt : array[0..40] of Char
    = 'Text file (*.txt)'#0'*.txt'#0;


PROCEDURE  WndProC.wmLButtonDown(var msg : TMessage);
VAR  ofn : TOpenFileName;
     path,fispec,filter,title : array[0..255] of Char;
     id,fiofs,extofs : Word; P : PChar;
BEGIN
  StrCopy(path,curfi);  P := StrRScan(path, '\');
  if  P <> nil  then  P[0] := #0;
  StrCopy(fispec, P + 1);  P := @filter;
  for  id := 1 to numfld  do
    begin  StrCopy(P, filtarray[id]);  P := StrEnd(P);
           P := P + 1  end;   P[1] := #0;

  FillChar(ofn, SizeOf(ofn), #0);
  with  ofn  do
    begin  lStructSize := SizeOf(TOpenFileName);
           hWndOwner   := HWindow;
           lpstrFilter := filter;
           lpstrCustomFilter := userflt;
           nMaxCustFilter := 40;
           nFilterIndex   := oldfltr;
           lpstrInitialDir := path;
           lpstrFile   := fispec;
           nMaxFile    := SizeOf(fispec);
           lpstrFileTitle := title;
           nMaxFileTitle := SizeOf(title);
           lpstrTitle  := 'Open BMPs file';
           Flags := ofn_HideReadOnly or ofn_CreatePrompt;
           lpstrDefExt := 'bmp' end;
  if  not GetOpenFileName(ofn)  then  Exit;
  oldfltr := ofn.nFilterIndex;
  fiofs := ofn.nFileOffset;  extofs := ofn.nFileExtension;
  StrCopy(curfi, fispec);
WriteLn('Path   = ', path,'  filter = ',oldfltr);
WriteLn('Fispec = ',fispec,'  title = ',title);
WriteLn('Fi ofset = ',fiofs,'  Ext ofset = ',extofs);
WriteLn('--------------------------');
  StrCopy(fispec,thetitle);  StrCat(fispec, curfi);
  SetWindowText(HWindow, fispec);
END;


(* ------------------  TheAppC Class  ---------------- *)
```

ประกอบกับค่าที่ให้ในโปรแกรมของกิจกรรม wmLButtonDown ตามลิสติงที่ 1 ด้วย กล่าวคือ

– lStructSize เป็นขนาดของตัวมันเอง (ไม่ทราบเหมือนกันว่าทำไมต้องให้ขนาดของตัวเองแก่ตัวเอง)

– hWndOwner หน้าต่างแม่

– lpstrFilter ถึง nFilterIndex ขอนำไปรวมกล่าวในหัวข้อ ฟิลเตอร์

– lpstrInitialDir เพื่อให้พาธเนม คือ ไดรฟ์ และไดเรคทอรีที่ต้องการให้ผู้ใช้เลือกไฟล์

– lpstrFile เพื่อให้ชื่อไฟล์ เช่น ในครั้งแรกอาจให้เป็น *.bmp และผู้ใช้เลือก tartan.bmp เมื่อให้ชื่อไฟล์นี้ในครั้งที่สองจะแสดงชื่อไฟล์นี้ในกรอบรายการไฟล์ เรื่องนี้ผมชอบมาก เพราะไม่ต้องไปเริ่มต้นที่ต้นรายการใหม่ทุกครั้ง และสำหรับในเที่ยวกลับ lpstrFile จะรับค่าไฟล์สเปกประกอบด้วยไดรฟ์ ไดเรคทอรี และชื่อไฟล์กลับไป ซึ่งในการให้ชื่อไฟล์หากเราให้เป็นไฟล์สเปกแทนจะทำให้ไม่ใช้ค่าใน IpstrInitialDir (แต่จะแสดงไฟล์สเปกที่กรอบ File Name ทำให้ดูแปลกไป)

– IpstrFileTitle เพื่อรับชื่อไฟล์กลับไป จุดประสงค์คือ เพื่อให้นำไปแสดงที่ไตเติลบาร์ของหน้าต่างแม่

– lpstrTitle ข้อความให้แสดงที่ไตเติลบาร์ของไดอะล็อกบ็อกซ์ ถ้าให้เป็น nil จะแสดงข้อความมาตรฐาน คือ Open หรือ Save As

– Flags เพื่อดำเนินการในรายละเอียดของไดอะล็อกบ็อกซ์ (ดูได้จาก ofn) ค่าที่ให้หมายถึง ไม่ต้องแสดงกรอบ ReadOnly และให้แสดงพรอมเตอร์คือเมสเสจบ็อกซ์เตือนผู้ใช้เมื่อป้อนชื่อไฟล์ที่ไม่มีในไดเรคทอรีนั้น

– nFileOffset และ nFile-

ลิสติงที่ 1 (ต่อ)

```
TYPE
  TheAppC = OBJECT(TApplication)
    procedure  InitMainWindow; virtual;
            END;


PROCEDURE  TheAppC.InitMainWindow;
BEGIN
  MainWindow := New(WndProCP, Init(nil,thetitle));
END;

(* --------------------  MAIN PROGRAM  ----------------- *)

VAR   app :  TheAppC;

BEGIN
  app.Init(appname);
  app.Run;
  app.Done;
END.
```

Extension คือ ตำแหน่งของชื่อไฟล์และนามสกุลใน lpstrFile ที่ส่งกลับมา

– lpstrDefExt เป็นนามสกุลที่จะให้ใช้เมื่อผู้ใช้ไม่ป้อน

– ฟิลด์ที่ขึ้นต้นด้วย nMax คือ ขนาดของบัฟเฟอร์ที่ใช้รับข้อความตามชื่อที่ต่อท้าย

ฟิลด์นอกนั้นจะใช้เมื่อเราสร้างรูปแบบ (template) ของไดอะล็อกบ็อกซ์เอง ซึ่งผมเห็นว่าไม่ควรใช้วิธีนี้ นอกจากนี้ก็มีเรื่องที่จะขอนำไปกล่าวในหัวข้อ ค่าแฟลกที่น่าสนใจ

ฟิลเตอร์

ในวินโดวส์เรียกชื่อไฟล์ซึ่งประกอบด้วยไวลด์การ์ด และนามสกุล เช่น *.PAS ว่าฟิลเตอร์ ในความหมายที่จะกรองคือ นำแต่ไฟล์ที่มีนามสกุลนี้มาแสดงในกรอบรายการไฟล์ และได้ออกแบบให้มีกรอบคอมโบบ็อกซ์ คือ List Files of Type เพื่อแสดงรายการฟิลเตอร์ โดยให้เราผ่านค่าไปทาง lpstrFilter ค่าที่ผ่านไปมีลักษณะเป็นอะเรย์ของสตริง โดยจัดเป็นคู่ ๆ คู่หนึ่ง ๆ จะประกอบด้วยสตริงข้อความที่ต้องการให้แสดงในกรอบรายการฟิลเตอร์ เช่น Text Files แต่นิยมเขียนเป็น Text Files (*.txt) กับข้อความที่จะให้แสดงที่กรอบ File Name เช่น *.txt หากต้องการหลายนามสกุลให้ใช้เครื่องหมาย ; คั่น เช่น *.txt; *.bak

อะเรย์ของฟิลเตอร์นี้จะต้องปิดด้วยอักขระนัลล์ (null) คือ #0 อีกครั้งหนึ่ง เช่น มีฟิลเตอร์อยู่ 5 คู่จะมีอักขระนัลล์อยู่แล้ว 10 ตัว จะต้องปิดท้ายอีก 1 ตัวเป็น 11 ตัว และสองตัวสุดท้ายจะอยู่ติดกัน ในโปรแกรมจึงต้องนำค่าที่ให้ไว้เป็นค่าคงตัว filt array มาให้แก่ filter เพื่อให้เป็นไปตามนี้

ค่าฟิลเตอร์ตามที่กล่าวมานี้นับลำดับที่เป็นคู่ เริ่มจากคู่ที่ 1 (ไม่ใช่ 0) เช่นมี 5 คู่ก็จะเป็นคู่ที่ 1 ถึงคู่ที่ 5 เป็นค่าหมายเลขฟิลเตอร์ที่ต้องให้แก่ฟิลด์ nFilterIndex เพื่อให้นำข้อความแรกของคู่นี้ไปแสดงที่กรอบข้อความของกรอบคอมโบ จึงต้องให้ตรงกับนามสกุลในชื่อไฟล์ที่ให้แก่ lpstrFile ด้วย

ดังเช่นในโปรแกรมนี้ได้ให้ชื่อไฟล์เป็น *.bmp จึงให้หมายเลขฟิลเตอร์เป็น 1 แก่ oldfltr เพื่อให้แก่ nFilterIndex อีกทอดหนึ่ง จะได้ผลการรันตามรูปที่ 1 เช่นกันหากผู้ใช้เลือกฟิลเตอร์ใหม่ จะได้หมายเลขฟิลเตอร์ใหม่คืนมาทาง nFilterIndex นำมาให้แก่ oldfltr เก็บไว้สำหรับการเรียกใช้ GetOpenFileName ในครั้งต่อไป

อีกกรณีหนึ่งคือ ผู้ใช้เปลี่ยนนามสกุลของไฟล์ใหม่ ซึ่งกระทำได้หลายวิธี เท่ากับเป็นการเปลี่ยนฟิลเตอร์ใหม่ และถ้าเป็นฟิลเตอร์ที่ไม่ได้กำหนดไว้คือไม่มีในกรอบรายการฟิลเตอร์ หากต้องการเพิ่มในกรอบรายการให้ทำรายการฟิลเตอร์นี้เพียง 1 คู่ แต่ต้องมีหน่วยความจำรองรับไม่น้อยกว่า 40 ไบต์ ดังที่ทำเป็นค่าคงตัวให้แก่ userflt และนำไปให้ lpstrCustomFilter ฟิลเตอร์ใหม่นี้จะมีค่าหมายเลขฟิลเตอร์เป็น 0 จึงต้องให้ค่านี้แก่ nFilterIndex ด้วย

ค่าที่ให้แก่ userflt เป็นเพียงตัวอย่างการให้ค่าฟิลเตอร์ 1 คู่เท่านั้น ที่จริงจะต้องตรวจจากไฟล์สเปกที่ผู้ใช้เลือก ดังจะแสดงในตัวอย่างต่อไป

ขนาดของบัฟเฟอร์และสแต็ก

บัฟเฟอร์ที่จะใช้รับค่าจากฟังก์ชัน GetOpenFileName และฟังก์ชันอื่น ๆ ในกลุ่มนี้ด้วยต้องมีขนาดใหญ่กว่าปกติเป็นพิเศษ ดังเช่น การรับค่าไฟล์สเปกจาก lpstrFile ถ้าใช้สตริงขนาด 80 ไบต์ก็นับว่ายาวมากแล้ว แต่ก็ไม่เพียงพอต้องเพิ่มเป็น 128 ไบต์ ก็ไม่เพียงพออีก ต้องใช้เป็น 256 ไบต์ตามตัวอย่างในคู่มือของวินโดวส์

อีกประเด็นหนึ่งคือ การใช้ไดอะล็อกบ็อกซ์กลางเหล่านี้ จะต้องกำหนด

เนื้อที่ของสแต็กไว้ไม่น้อยกว่า 8 กิโลไบต์ (เท่ากับที่กำหนดไว้ในเทอร์โบปาสคาลตามปกติ)

โปรแกรมทดลอง

โปรแกรมตามลิสติงที่ 1 คือโปรแกรมสำหรับให้ทดลองตามที่กล่าวมาครับ หากข้อความตอนใดผมกล่าวไว้ไม่ชัดเจน ขอให้ทดลองเปลี่ยนค่าดู โดยที่ผมได้ให้แสดงผลที่เป็นเอาต์พุตทั้งหมดด้วยคำสั่ง Write ผลจึงปรากฏที่หน้าต่างของ WinCrt เป็นวิธีหนึ่งที่ใช้เก็บผลการรันไว้ศึกษา

สร้างฟังก์ชันซ้อนฟังก์ชัน

การใช้งานฟังก์ชัน GetOpenFileName ตามที่กล่าวมาดูแล้วค่อนข้างจะไม่กะทัดรัด เพราะต้องผ่านค่าต่าง ๆ มากมาย ซึ่งความประสงค์ของผู้เขียนโปรแกรมแท้ ๆ คงต้องการแค่เพียงผ่านค่าไฟล์สเปกไปให้ และรับค่าไฟล์สเปกที่ผู้ใช้เลือกกลับมา กับจะมีข้อแตกต่างกันในเรื่องของฟิลเตอร์ และข้อความที่จะให้แสดงเป็นแคปชันของไดอะล็อกบ็อกซ์

ประกอบกับทั้ง GetOpenFileName และ GetSaveFileName ต่างก็ใช้แบบข้อมูล TOpenFileName ในการผ่านค่าทั้งคู่ จึงควรทำเป็นฟังก์ชันครอบฟังก์ชันทั้งสองนี้ไว้ เพื่อให้สามารถใช้ฟังก์ชันคู่นี้ได้โดยง่ายในการใช้งานธรรมดาทั่ว ๆ ไป ดังฟังก์ชัน GetFiName ในลิสติงที่ 2

โดยให้ผ่านค่ามาให้ตามลำดับดังนี้

– awnd เป็นค่าแฮนเดิลของหน้าต่างแม่

– filter สตริงที่จะทำเป็นค่าฟิลเตอร์

ลิสติงที่ 2

```
PROCEDURE  CheckFilter(filtr,fiext : PChar; var ipair : Word);
VAR   icnt : Word;  P, Pf : PChar;  stp,ch : Char;
BEGIN
  P := StrEnd(filtr) - 1;  stp := P[0];  P := filtr;
  ipair := 0;  icnt := 1;  Pf := P;
  while  P[0] <> #0  do
    begin
    if  P[0] = stp  then
      begin  P[0] := #0;
             if  icnt mod 2 = 0  then
               if  lstrcmpi(fiext, Pf) = 0  then
                 ipair := icnt div 2;
             Pf := P + 3;  icnt := icnt + 1  end;
    P := P + 1;
    end;
  P[0] := #0;
END;


FUNCTION   GetFiName(awnd : HWnd; filter, capt,filespec : PChar;
                     islode : Boolean) : Boolean;
VAR  ofn : TOpenFileName;
     finame,path,filtr : array[0..255] of Char;
     usrflt : array[0..40] of Char;
     id,ipair : Word; Pext,Pusr : PChar;  res : Boolean;  ch : Char;
BEGIN
  StrCopy(path, filespec);  Pext := StrRScan(path, '\');
  if  Pext <> nil  then  Pext[0] := #0;
  StrCopy(finame, Pext + 1);   Pext := StrRScan(finame, '.') + 1;

  StrCopy(filtr, filter);  CheckFilter(filtr, Pext, ipair);
  if  ipair = 0  then
    begin  StrCopy(usrflt,'File (*.');  StrCat(usrflt, Pext);
           StrCat(usrflt,')');  Pusr := StrEnd(usrflt) + 1;
           StrCopy(Pusr, '*.');  StrCat(Pusr, Pext);
           Pusr := StrEnd(Pusr);  Pusr[1] := #0;
           Pusr := @usrflt  end
  else  Pusr := nil;

  FillChar(ofn, SizeOf(ofn), #0);
  with  ofn  do
    begin  lStructSize := SizeOf(TOpenFileName);
           hWndOwner   := awnd;
           lpstrFilter := filtr;
           lpstrCustomFilter := Pusr;
           nFilterIndex:= ipair;
           lpstrInitialDir:= path;
           lpstrFile   := finame;
           nMaxFile    := SizeOf(finame);
           lpstrTitle  := capt;
           Flags := ofn_HideReadOnly or ofn_CreatePrompt
                    or ofn_OverWritePrompt;
           lpstrDefExt := Pext  end;
  if  islode  then  res := GetOpenFileName(ofn)
  else  res := GetSaveFileName(ofn);
  if  res  then  GetFiName := true
  else begin  GetFiName := false;  exit  end;
  StrCopy(filespec, finame);
  if  islode  then
    begin
    GetWindowText(awnd,finame,79);  Pext := StrRScan(finame, '-');
    if  Pext <> nil  then
      begin  Pext[1] := ' ';  Pext[2] := #0  end
    else  finame[0] := #0;
    StrCat(finame, filespec);
    SetWindowText(awnd, finame)
    end;
END;
```

ลิสติงที่ 3

```
PROCEDURE  WndProC.wmLButtonDown(var msg : TMessage);
VAR  fispec : array[0..255] of Char;
BEGIN
  if  not  GetFiName(HWindow,'Bitmaps file (*.BMP)¦*.bmp¦' +
                         'Pascal file (*.PAS *.BAK)¦*.pas;*.bak¦' +
                         'Document file (*.DOC)¦*.doc¦' +
                         'Header file (*.H)¦*.h¦' +
                         'All files (*.*)¦*.*¦',
                         nil, curfi, true)  then  Exit;
  StrCopy(fispec,thetitle);  StrCat(fispec, curfi);
  SetWindowText(HWindow, fispec);
END;
```

– capt ข้อความที่จะให้แสดงที่ไตเติลบาร์ของไดอะล็อกบ็อกซ์ หากให้ค่าเป็น nil จะได้ไตเติลเป็น Open หรือ Save As

– filespec คือไฟล์สเปกตามที่กล่าวมา

– islode หากให้เป็นจริงจะเป็นการเรียกใช้ GetOpenFileName หากให้เป็นเท็จจะเรียกใช้ GetSaveFileName

การเขียนโปรแกรมของฟังก์ชันนี้คงเป็นไปตามที่กล่าวมา ที่แตกต่างไปจากลิสติงที่ 1 คือ

ค่าฟิลเตอร์

ให้ผ่านค่ามาเป็นสตริง ขอให้ดูในลิสติงที่ 3 (โดยสมมติว่าได้แก้โปรแกรมของ wmLButtonDown ในลิสติงที่ 1 ให้เป็นดังในลิสติงที่ 3) ค่าฟิลเตอร์ทั้งหมดนั้นคือสตริงเดียวกัน แต่เนื่องจากต้องการขึ้นบรรทัดใหม่จึงใช้เครื่องหมาย + ต่อค่าคงตัวสตริง แต่ละบรรทัดหรือฟิลเตอร์คู่หนึ่ง โดยให้ปิดท้ายข้อความแต่ละท่อนด้วยเครื่องหมายใดก็ได้ ตามตัวอย่างนี้ใช้ขีดตั้ง

ซึ่งใน GetFiName จะให้ CheckFilter ทำเป็นอะเรย์ของฟิลเตอร์ โดยอ่านเครื่องหมายที่ปิดท้ายข้อความให้แก่ stp แล้วตรวจในสตริง เปลี่ยนเครื่องหมายนี้เป็นอักขระนัลล์คือ #0 ทุกแห่ง และปิดท้ายสุดด้วย #0 อีกครั้ง

ค่าหมายเลขฟิลเตอร์

เนื่องจากทำให้เป็นฟังก์ชันจึงเก็บค่าใน nFilterIndex ไว้ไม่ได้ จะต้องหาใหม่ โดยนำนามสกุลในไฟล์สเปกให้ CheckFilter ตรวจเทียบกับค่าฟิลเตอร์ เมื่อตรงกับคู่ใดจะให้หมายเลขฟิลเตอร์ของคู่นั้นแก่ ipair นำไปให้แก่ nFilterIndex อีกทอดหนึ่ง หากไม่ตรงกับคู่ใดจะให้ ipair เป็น 0

ฟิลเตอร์ของผู้ใช้

เมื่อผู้ใช้เปลี่ยนนามสกุลเท่ากับเปลี่ยนฟิลเตอร์ และถ้าไม่มีในรายการฟิลเตอร์ จะเป็นฟิลเตอร์ใหม่ของผู้ใช้ได้ ค่า ipair เป็น 0 ใน GetFiName จึงนำค่าใน ipair มาตรวจ สมมติว่าผู้ใช้เลือกนามสกุลใหม่เป็น DLL คำสั่งทั้งหมดนั้นจะเท่ากับให้ค่าต่อไปนี้แก่ usrflt คือ

```
'File (*.DLL)'#0'*.dll'#0;
```

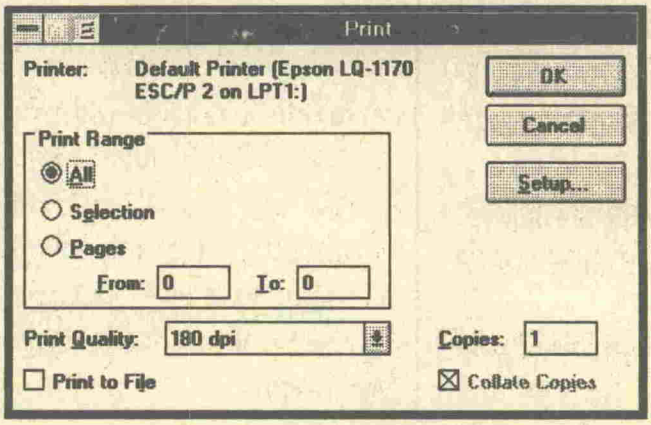
การป้อนโปรแกรม

ขอให้ป้อนโปรแกรมตามลิสติงที่ 2 เก็บไว้ในยูนิต UFiDlg คือไฟล์ UFIDIG.PAS (ตามที่กล่าวติดต่อมาหลายฉบับแล้ว) เสร็จแล้วให้คอมพายล์เป็นไฟล์ .TPU รวมทั้งต้องคอมพายล์ UPRINT1.PAS ใหม่ด้วยครับ

PrintDlg

อีกไดอะล็อกบ็อกซ์หนึ่งคือ PrintDlg สำหรับให้ผู้ใช้ปรับแต่งเครื่องพิมพ์ เมื่อเรียกใช้ฟังก์ชันนี้จะแสดงไดอะล็อกบ็อกซ์ดังรูปที่ 3 ถ้าผู้ใช้เลือกปุ่ม Setup จะแสดงไดอะล็อกบ็อกซ์ Print Setup ดังที่เห็น เมื่อเลือกรายการนี้ของแอพพลิเคชันต่าง ๆ และมีไดอะล็อกบ็อกซ์ Options ต่อไปอีก

การใช้งานก็เช่นเดียวกับไดอะ-



รูปที่ 3
ไดอะล็อกบ็อกซ์กลางของฟังก์ชัน PrintDlg

ล็อกบ็อกซ์กลางทั้งหลายดังที่กล่าวมา คือ ต้องให้ค่าแก่เรคอร์ดตามแบบข้อมูลที่จัดไว้ของฟังก์ชัน PrintDlg คือ TPrintDlg คงเป็นเช่นเดียวกับของไฟล์ คือมีฟิลด์มากมาย แต่ในการใช้งานตามปกติ และโดยเฉพาะบิตแม็พเหลือฟิลด์ที่จะต้องให้ และอ่านค่าน้อยมาก ขอแสดงตัวอย่างการใช้เลยครับ ตามลิสติงที่ 4 กล่าวคือ

ลิสติงที่ 4

```
PROCEDURE  WndProC.cmPrint(var msg : TMessage);
VAR   aprn : PrinterCP;   num,swid,shi : Integer;
      pd : TPrintDlg;     txt : PChar;
BEGIN
  if  mainBMP = 0  then  Exit;  num := 2;
  if  Application^.ExecDialog(New(EditNumDlgCP,
        Init(@self, 'Enter number of time to enlarge.',
             'PRINTING INFORMATION','i',2,num)))
       = id_Cancel  then  Exit;
  if  num < 1 then  Exit;

  FillChar(pd, SizeOf(TPrintDlg), #0);
  pd.lStructSize := SizeOf(TPrintDlg);
  pd.HWndOwner   := HWindow;
  pd.Flags       := pd_ReturnDC;

  if  not PrintDlg(pd)  then  Exit;
  txt := 'Test Doc';
  Escape(pd.hDC, StartDoc, StrLen(txt),txt,nil);

  PrnDspBmps(HWindow, mainBMP, pd.hDC,
                200,200,mainwid*num,mainhi*num, false,
                0,0,mainwid,mainhi);

  Escape(pd.hDC, NewFrame, 0, nil, nil);
  Escape(pd.hDC, EndDoc, 0, nil, nil);
  DeleteDC(pd.hDC);
  if  pd.hDevMode <> 0  then GlobalFree(pd.hDevMode);
  if  pd.hDevNames <> 0  then GlobalFree(pd.hDevNames);
END;
```

1. ได้กำหนดตัวแปร pd เป็นตัวแปรเรคอร์ดแบบ TPrintDlg

2. โปรแกรมท่อนแรกยังไม่เกี่ยวกับ pd ครับ เป็นท่อนที่แสดงไดอะล็อกบ็อกซ์ให้ผู้ใช้ป้อนจำนวนเท่าที่ต้องการขยายภาพในการพิมพ์

3. เริ่มเรื่องของ pd โดยที่ให้ค่าเรคอร์ดนี้เป็นศูนย์ทุกไบต์ เพื่อที่จะให้ฟิลด์ที่เราไม่ได้ให้ค่ามีค่าเป็น 0 หรือ nil เสร็จแล้วได้ให้ค่าแก่ฟิลด์ต่าง ๆ และมีฟิลด์ที่ควรนำมากล่าวถึงดังนี้ คือ

– lStructSize เพื่อให้ขนาดของตัวมันเอง

– hDevMode เป็นพอยน์เตอร์ไปยังเรคอร์ดต่อไปอีก กำหนดโหมดการทำงานของเครื่องพิมพ์ แต่ความจริงก็คือรายละเอียด เช่น ขนาดของกระดาษ เป็นต้น เมื่อให้เป็น nil วินโดวส์จะใช้ค่าตามที่ผู้ใช้ติดตั้งไว้

– hDevName เป็นพอยน์เตอร์ไปยังเรคอร์ดอีกเช่นกัน กำหนดชื่อเครื่องพิมพ์ ไดรเวอร์ และพอร์ต ดังที่ใช้ในการกำเนิดพรินต์ดีซี (ไมโครฯ 93) เมื่อให้ค่าเป็น nil วินโดวส์จะใช้ค่าที่ติดตั้งอยู่ในขณะนั้น

– Flags คือรายละเอียดที่เราจะให้ดำเนินการ ค่าที่ให้หมายถึงขอค่าพรินต์ดีซี

– hDC เมื่อขอพรินต์ดีซีจะได้ค่าที่ขอในฟิลด์นี้

– nCopies จำนวนสำเนาที่ผู้ใช้ต้องการ

4. ให้แสดงไดอะล็อกบ็อกซ์ด้วยการเรียกใช้ฟังก์ชัน PrintDlg

5. โปรแกรมส่วนอื่นคงเขียนตามปกติ ที่เป็นพิเศษคือ ไม่ต้องให้กำเนิดพรินต์ดีซีอีก นำค่าที่ขอมาใช้ได้โดยตรง

ฟิลด์และแฟลกที่ผมไม่ได้กล่าวถึงจะเกี่ยวข้องกับการพิมพ์ข้อความ และการใช้ไดอะล็อกบ็อกรีซอร์สของเราเอง ขอให้ดูรายละเอียดได้ใน Help ครับ

เมื่อผู้ใช้เลือก หรือป้อน หรือเซตค่าในไดอะล็อกบ็อกซ์ วินโดวส์จะเป็นผู้รับไปดำเนินการครับ เช่น เมื่อผู้ใช้เลือกแบบการพิมพ์เป็น Landscape ก็จะเป็นการพิมพ์ทางขวางโดยที่เราไม่ต้องรับทราบ

โปรแกรมนี้สามารถนำไปใช้แทนโปรแกรมของกิจกรรมเดียวกันนี้คือ cmPrint ตามที่จะกล่าวต่อไปในลิสติงที่ 5 และเมื่อใช้โปรแกรมนี้ไม่ต้องกำหนดใช้ยูนิต uprint1

การนำไปใช้ในโปรแกรม

คือโปรแกรมจับภาพใส่หน้าต่างตามที่จะเล่าในฉบับหน้า ผมตั้งชื่อโปรแกรมไว้ว่า ScrnCaptor2 ที่ต้องเป็น 2 เพราะผมเตรียม 1 ไว้แล้ว เกิดเปลี่ยนใจใช้ไดอะล็อกบ็อกซ์กลางตามที่กล่าวมาและในฉบับนี้เป็น 2a ด้วย เพราะทำไแต่เพียงการอ่าน-เขียนภาพบิตแม็พกับไฟล์ นำออกแสดงทางจอภาพและเคองพิมพ์ ยังไม่เกี่ยวกับการ

จับภาพแต่อย่างใด

เนื่องจากโปรแกรมยาวกว่าคำอธิบาย ผมจึงขอลงติดต่อกัน ในลิสติงที่ 5 มีเรื่องที่จะขอให้คำเนินการและคำอธิบายดังนี้

รีซอร์ส

ขอให้สร้างเมนูรีซอร์สรายละเอียดตามรูปที่ 4 ให้ชื่อว่า SCRNCAP2_MENU เก็บไว้ในไฟล์ SCRNCAP2.RES เพื่อใช้กับโปรแกรมนี้ครับ

แนะนำตัวแปร

ในการกำหนดสกุล WndProC ได้กำหนดรายละเอียดของสกุลนี้คือตัวแปรไว้มากพอสมควร และได้ให้ค่าเริ่มต้นไว้ใน Init ซึ่งจะขอนำมาแนะนำดัง

ลิสติงที่ 5

```
PROGRAM  ScrnCaptor2a;  { scrcap2a.pas }
USES   commdlg,WinTypes,WinProcs,WObjects,Strings,
       WinCrt,ufidlg,uprint1;

{$R scrncap2.res}

CONST  appname  = 'scrncapt';
       thetitle = 'SCREEN CAPTURE - ';
       menursrc = 'SCRNCAP2_MENU';
       iconrsrc = 'SCRNCAP2_ICO';


  cm_New    = 101;  cm_FiOpen = 102;
  cm_Save   = 103;  cm_FiSaveAs = 104;  cm_PutBMP = 106;
  cm_Print = 107;   cm_PrnSet = 108;
  cm_GetMark = 151;  cm_GetMarkBMP = 152;
  cm_GetClpbrd  = 153;
  cm_Clear = 201;   cm_Mark    = 202;     cm_ClrMark = 203;
  cm_Copy  = 204;   cm_Cut     = 205;     cm_Paste   = 206;
  cm_PasteAtMark = 207;  cm_PasteOK = 208;
  cm_CopyTo = 209; cm_PasteFrom = 210;
  cm_ZoomIn = 51 ; cm_ZoomOut   = 52;
  cm_CaptHide = 301;  cm_CaptShow = 302;  cm_ScrCopy = 303;


TYPE

  CaptModeT = (nocapt, hhde, sshw, mmrk, mrkok);

  WndProCP = ^WndProC;
  WndProC = OBJECT(TWindow)
    ldefi,svefi : array[0..79] of Char;
    mainBMP, scndBMP  : HBitmap;
    srcx,srcy,srcwid,srchi : Integer;
    dstx,dsty,dstwid,dsthi : Integer;
    sttpnt,endpnt : TPoint;
    mainwid, mainhi : LongInt;
    scndwid, scndhi : LongInt;
    cpyflg,nopaint,mrkon : Boolean;
    cptmde         : CaptModeT;
constructor  Init(apar : PWindowsObject; atitle : PChar);
destructor   Done; virtual;
procedure  Paint(paintdc : HDC;
             var paintInfo : TPaintStruct); virtual;
procedure  cmPrint(var msg : TMessage);
             virtual cm_First + cm_Print;
procedure  cmFiOpen(var msg : TMessage);
             virtual cm_First + cm_FiOpen;
procedure  cmFiSaveAs(var msg : TMessage);
             virtual cm_First + cm_FiSaveAs;
procedure  BMPtoScrn(adc : HDC; dx,dy,dwid,dhi : Integer;
                     srceBMP : HBitmap; sx,sy,swid,shi : Integer);
procedure  SetDefault;
         END;
```

ลิสติงที่ 5 (ต่อ)

```
(* ------------------------------------------------ *)

procedure  SwapPoint(var frst,scnd : Integer);
var   tem : Integer;
begin
  if  frst > scnd  then
    begin  tem := frst;  frst := scnd;  scnd := tem  end;
end;


(* ------------------- WndProC Routine --------------- *)


CONSTRUCTOR  WndProC.Init(apar : PWindowsObject; atitle : PChar);
BEGIN
  TWindow.Init(apar,atitle);
  attr.menu := LoadMenu(HInstance, menursrc);
  attr.x := 100;  attr.y := 50;  attr.w := 400;  attr.h := 300;
  attr.Style := Attr.Style or ws_VScroll or ws_HScroll;
  Scroller := New(PScroller, Init(@Self, 1, 1, 100, 100));
  StrCopy(ldefi, 'c:\tpw\owldemos\*.bmp');
  StrCopy(svefi, 'c:\tpw\owldemos\*.bmp');
  dstx := 0;  dsty := 0;  dstwid := 0;  dsthi := 0;
  srcx := 0;  srcy := 0;  mainBMP := 0;  scndBMP := 0;
  cpyflg := true;  nopaint := false;  cptmde := nocapt;
  mrkon := false;
END;



DESTRUCTOR  WndProC.Done;
VAR   txt : array[0..19] of Char;
BEGIN
  if  mainBMP <> 0  then  DeleteObject(mainBMP);
  if  scndBMP <> 0  then  DeleteObject(scndBMP);
  TWindow.Done;
END;


PROCEDURE  WndProC.Paint(paintdc : HDC;
                         var paintInfo : TPaintStruct);
VAR   arect : TRect;
BEGIN
  mrkon := false;
  if (mainBMP = 0) or nopaint  then  Exit;
  BMPtoScrn(paintdc, dstx,dsty,dstwid,dsthi,
            mainBMP, srcx,srcy,srcwid,srchi);
  GetClientRect(HWindow, arect);
  scroller^.SetRange(dstx+dstwid - arect.right,
                     dsty+dsthi - arect.bottom);
END;



PROCEDURE  WndProC.BMPtoScrn(adc : HDC; dx,dy,dwid,dhi : Integer;
                             srceBMP : HBitmap; sx,sy,swid,shi : Integer);
VAR   hOldObj : THandle;  srceDC : HDC;  ispaint : Boolean;
BEGIN
  if  adc = 0  then
    begin  ispaint := false;  adc := GetDC(HWindow)  end
  else  ispaint := true;
  srceDC := CreateCompatibleDC(adc);
  hOldObj := SelectObject(srceDC, srceBMP);
```

ลิสติงที่ 5 (ต่อ)

```
  if  ispaint  then  Rectangle(adc,dx-1,dy-1,dx+dwid+1,dy+dhi+1);
  if  cpyflg  then
    BitBlt(adc, dx,dy,dwid,dhi, srceDC, sx,sy, srcCopy)
  else
    StretchBlt(adc, dx,dy,dwid,dhi,
               srceDC, sx,sy,swid,shi, srcCopy);

  if  not ispaint  then  ReleaseDC(HWindow, adc);
  SelectObject(srceDC, hOldObj);
  DeleteDC(srceDC);
END;



PROCEDURE  WndProC.cmPrint(var msg : TMessage);
VAR   aprn : PrinterCP;   num,swid,shi : Integer;
BEGIN
  if  mainBMP = 0  then  Exit;  num := 2;
  if  Application^.ExecDialog(New(EditNumDlgCP,
        Init(@self, 'Enter number of time to enlarge.',
             'PRINTING INFORMATION','i',2,num)))
        = id_Cancel  then  Exit;
  if  num < 1 then  Exit;
  New(aprn,Init(HWindow));
  if  aprn^.prnok  then
   begin
    if  aprn^.StartADoc('SCREEN CAPTOR')  then
      PrnDspBmps(HWindow, mainBMP, aprn^.prnDC,
                 100,100,mainwid*num,mainhi*num, num <> 1,
                 0,0,mainwid,mainhi);
    aprn^.NewPage;  aprn^.EndADoc;
    end;
  Dispose(aprn,Done);
END;



PROCEDURE  WndProC.cmFiOpen(var msg : TMessage);
VAR   finme : array[0..79] of Char;
BEGIN
  if  not  GetFiName(HWindow,'Bitmaps file (*.BMP)¦*.bmp¦' +
                             'All files (*.*)¦*.*¦',
                             nil,ldefi, true)  then  Exit;
  if  mainBMP <> 0 then  DeleteObject(mainBMP);
  StrCopy(finme,ldefi);
  mainBMP := LoadBitmapFile(finme, mainwid, mainhi);
  if  mainBMP = 0  then
    begin  MessageBox(HWindow, 'Can not open','INFO', mb_OK);
           Exit  end;
  SetDefault;
  InvalidateRect(HWindow,nil,true);
END;



PROCEDURE  WndProC.cmFiSaveAs(var msg : TMessage);
VAR   finme : array[0..79] of Char;
BEGIN
  if  mainBMP = 0  then  Exit;
  if  not  GetFiName(HWindow,'Bitmaps file (*.BMP)¦*.bmp¦' +
                             'All files (*.*)¦*.*¦',
                             nil,svefi, false)  then  Exit;
```

ลิสติงที่ 5 (ต่อ)

```
  StrCopy(finme,svefi);
  if  SaveBitmapFile(finme, mainBMP) = 0  then
    MessageBox(HWindow, 'Can not save', 'INFO', mb_OK)
  else  MessageBox(HWindow, 'File Save OK', 'INFO', mb_OK);
END;


PROCEDURE  WndProC.SetDefault;
BEGIN
  dstx := 5;  dsty := 5;  dstwid := Integer(mainwid);
  dsthi := Integer(mainhi);
  srcx := 0;  srcy := 0;  srcwid := Integer(mainwid);
  srchi := Integer(mainhi);
END;


(* ------------------  TheAppC Class  ---------------- *)

TYPE
  TheAppC = OBJECT(TApplication)
    procedure  InitMainWindow; virtual;
            END;


PROCEDURE  TheAppC.InitMainWindow;
BEGIN
  MainWindow := New(WndProCP, Init(nil,thetitle));
END;

(* --------------------  MAIN PROGRAM  ----------------- *)

VAR   app :  TheAppC;

BEGIN
  app.Init(appname);
  app.Run;
  app.Done;
END.
```

นี้ครับ

– Idefi และ svefi สำหรับเก็บไฟล์สเปกเพื่อการอ่านและเขียนกับไฟล์ตามลำดับ ได้ให้ค่าเริ่มต้นไว้เป็นไดเรคทอรีของวินโดวส์ ซึ่งมีไฟล์ .BMP อยู่

– mainBMP สำหรับเก็บคือชี้ไปที่ภาพบิตแม็พ ซึ่งถือว่าเป็นภาพบิตแม็พของหน้าต่าง แต่ขนาดของภาพไม่ได้ขึ้นกับขนาดของหน้าต่าง มีค่าความกว้างและความสูงเก็บไว้ใน mainwid และ mainhi ตามลำดับ

– ในการแสดงภาพบิตแม็พทางหน้าต่าง ถือว่าตัวแผ่นภาพบิตแม็พซึ่งส่วนใหญ่จะได้แก่ mainBMP คือต้นทาง และหน้าต่างคือปลายทาง โดยให้ผู้ใช้เลือกได้ว่าจะนำส่วนใดของภาพไปแสดงตรงส่วนใดของหน้าต่าง ด้วยการให้ค่าแก่ตัวแปรในชุด srcx ถึง srchi กับชุด dstx ถึง dsthi ดังการให้ค่าเบื้องต้น (de fault) ใน SetDefault

สำหรับตัวแปรอื่นตามที่จะกล่าวต่อไปนี้ เมื่อนำมาใช้ครับ

อ่านจากไฟล์

เมื่อผู้ใช้เลือกรายการ File/Open เราจะได้รับเมสเสจ cm_FiOpen ได้จัดกิจกรรม cmFiOpen ไว้ตอบรับเมสเสจนี้ โดย

– เรียกใช้ GetFiName เพื่อจะใช้ไฟล์ไดอะล็อกบ็อกซ์ของวินโดวส์ให้ผู้ใช้เลือกไฟล์ที่ต้องการอ่าน

– นำชื่อไฟล์มาให้ LoadBitmapFile (อยู่ในยูนิต UFiDlg) โหลดบิตแม็พมาทำเป็นแผ่นภาพบิตแม็พให้แก่ mainBMP ซึ่งจะได้ค่าความกว้างและความสูงของภาพมาด้วย

– บังคับให้ไปทำงานที่เพนต์ด้วย InvalidateRect

เพนต์

การมาทำงานที่เพนต์คือ Paint เกิดจาก 2 กรณีคือ ในโปรแกรมบังคับให้มา เช่น เมื่อโหลดไฟล์ใหม่ หรือจับภาพมาใหม่ กับวินโดวส์ส่งมาเมื่อเริ่มต้นเปิดหน้าต่าง และเมื่อผู้ใช้เปลี่ยนขนาดของหน้าต่าง จุดประสงค์คือให้แสดงภาพในพื้นที่ใช้งาน (client area) ใหม่ ซึ่งในโปรแกรมนี้ได้

– ให้ BMPtoScrn นำภาพใน mainBMP ออกแสดงตามความต้องการดังกล่าวมา

– ให้ระยะเลื่อนแก่สโครลบาร์ใหม่ ซึ่งความจริงโปรแกรมส่วนนี้ควรอยู่ใน wmSize แต่เรามีโปรแกรมที่จะเปลี่ยนแปลงขนาดของภาพอยู่ในที่หลายแห่ง จึงนำมารวมไว้ที่ Paint

จากบิตแม็พสู่จอภาพ

โปรแกรมของกิจกรรม BMPtoScrn จะเป็นผู้นำภาพบิตแม็พที่ผ่านมาให้แก่ screBMP ออกแสดงทางจอภาพคือในหน้าต่าง โดยที่ผู้ที่เรียกใช้ไม่ใช่แค่ Paint เท่านั้น จึง

– ตรวจว่าถ้าไม่ใช่ Paint จะต้องขอดีซี

– ถ้าเป็น Paint ให้ตีกรอบด้วยเพื่อให้เห็นขอบเขตของแผ่นภาพ

```
&File
  &New              101
  &Open...          102
  &Save             103
  Save &As...       104
    ------------
  &Get
    &Mark           151
    &BMP mark       152
    &Clipboard      153
  Put &BMP          106
    ------------
  &Print            107
  Printer se&tup... 108
&Edit
  &Clear            201
    ------------
  &Mark             202
  C&lear Mark       203
    ------------
  C&opy             204
  C&ut              205
    ------------
  &Paste            206
  Paste at &mark    207
  Paste &OK         208
    ------------
  Copy &to...       209
  Paste &from...    210
&Capture
  &Hide             301
  &Show             302
    ------------
  Screen Copy       303
Zoom &In            51
Zoom &out           52
```

รูปที่ 4 รายละเอียดของเมนูรีซอร์ส SCRNCAP2_MENU

– การนำออกแสดง ให้ตรวจ cpyflg ถ้าเป็นจริงคือ ตามปกติจะเป็นการลอกแบบตรงตัวด้วย BitBlt หากเป็นเท็จจะลอกแบบย่อขยายด้วย StretchBlt ซึ่งถ้าพื้นที่ปลายทางใหญ่กว่าต้นทาง จะเป็นการขยาย หากน้อยกว่าจะเป็นการย่อ

เขียนกับไฟล์

เมื่อให้อ่านไฟล์แล้วก็ควรให้เขียนกับไฟล์ได้ด้วย โดยให้ผู้ใช้เลือกรายการ File/Save As จะมาทำงานที่ cmfiSaveAs คือ

– ให้ GetFiName เรียกใช้ไดอะล็อกบ็อกซ์ Save As ของวินโดวส์ให้ผู้ใช้เลือกหรือป้อนชื่อไฟล์

– นำชื่อไฟล์มาให้ SaveBitmapFile (ใน ยูนิต UFiDlg) นำภาพบิตแม็พใน mainBMP ไปเก็บลงไฟล์นี้

งานพิมพ์

เพื่อให้ครบถ้วน ก็ต้องให้พิมพ์ได้ด้วยครับ ได้เตรียมรายการ File/Print ไว้ให้เลือก ซึ่งจะมาทำงานที่ cmPrint ซึ่งจะ

– ให้ผู้ใช้ป้อนจำนวนเท่าที่จะขยายภาพบิตแม็พในการพิมพ์ ค่าที่เหมาะสมคือ 2 เท่า ตามที่แสดงเป็นตัวอย่าง

– นำสกุล PrinterC ที่สร้างไว้ในยูนิต UPrint1 มาใช้ในการพิมพ์ (ตามที่กล่าวในฉบับก่อนครับ)

– ส่วนการพิมพ์ภาพบิตแม็พแท้ ๆ ได้เรียกใช้ PrnDspBmps (ในยูนิต UFiDlg) ค่า num <> 1 จะทำให้เกิดการขยายเมื่อค่านี้เป็นจริง

ซอฟต์เมล์

ฉบับนี้คงมีเท่านี้ก่อน สำหรับโปรแกรม ScrnCaptor2 ที่ผมฝากมาทางซอฟต์เมล์ เป็นโปรแกรมที่สมบูรณ์แล้วคือรวมทั้งที่จะกล่าวในอีก 2 ฉบับด้วยครับ

ขอแก้ที่ผิด

มีเรื่องที่ต้องขออภัยอีกตามเคย คือ มีที่ผิดอยู่ในโปรแกรมของฟังก์ชัน SaveBitmapFile ในลิสติงที่ 2.4 ในไมโครฯ 92 ครับ เป็นโปรแกรมที่ผมลอกมาจากไฟล์ BITMAPS.PAS ในชุดของ TPW 1.5

ที่ผิดคือตรงที่ใช้ฟังก์ชัน GetDIBits ได้นำค่าขนาดของภาพคือ pctsi มาให้เป็นค่าจำนวนเส้นสแกนไลน์ ค่าที่ควรจะนำมาใช้คือ abmp.bmHeight ขอให้แก้โปรแกรมส่วนนี้ให้เป็นดังรูปที่ 5 (และขอเพิ่มรูปที่ 6 ครับ) เสร็จแล้วจะต้องคอมพายล์ยูนิต UFiDlg กับ UPrintl ใหม่

```
UNIT  UFIDLG;
USES  Objects,WinTypes,WinProcs,OWindows,
      ODialogs,Strings,commdlg;


UNIT   UPrint1;
USES   Objects,WinTypes,WinProcs,OWindows,
       ODialogs,Strings,ufidlg;


PROGRAM  ScrnCaptor2c;
USES   WinTypes,WinProcs,OWindows,
       Strings,commdlg,ufidlg,uprint1;
```

รูปที่ 5

```
if GetDIBits(adc, hBmp, 0, abmp.bmHeight, buf,
             pInfo^, dib_RGB_Colors) <> 0 then
```

รูปที่ 6